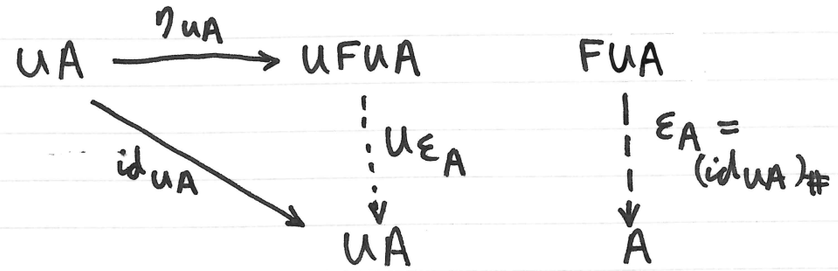
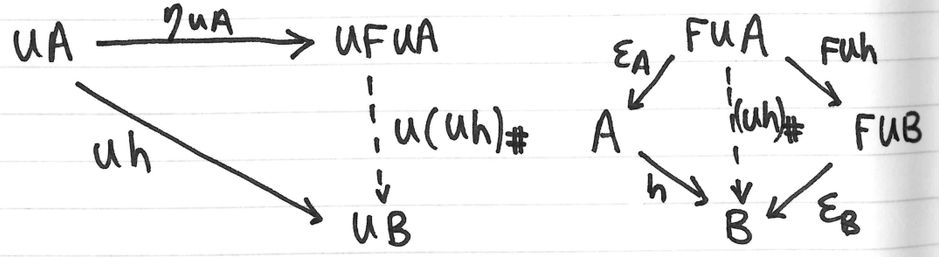
Next we define $\varepsilon_A : F(UA) \longrightarrow A$ by $\varepsilon_A = (id_{UA})\#$, the unique arrow



s.t. $id_{UA} = U\varepsilon_A \cdot \eta_{UA}$. The triangle here gives immediately the law $U\varepsilon \cdot \eta U = Id_U$.

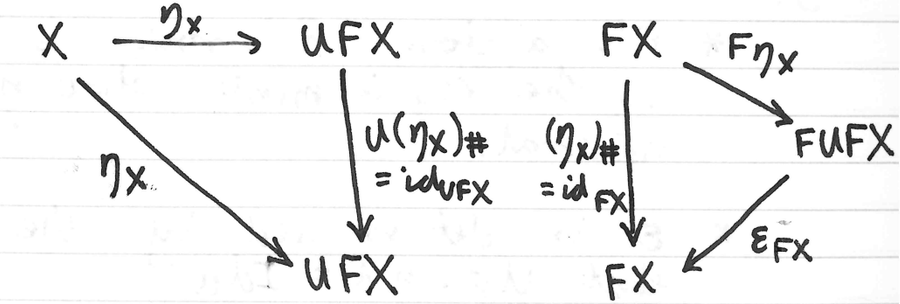
For the naturality of ε we need to prove for each $h : A \longrightarrow B$ in $\mathcal{A}$ the equality of the two arrows $\varepsilon_B \cdot FUh$ and $h \cdot \varepsilon_A$ from FUA to A.



In fact, they are both equal to $(Uh)\#$ because

$$
\begin{aligned}
& U(h \cdot \varepsilon_A) \cdot \eta_{UA} \\
= \; & \{ U \text{ a functor} \} \\
& Uh \cdot U\varepsilon_A \cdot \eta_{UA} \\
= \; & \{ U\varepsilon \cdot \eta U = Id_U \} \\
& Uh \cdot id_{UA} \\
= \; & \{ id \} \\
& Uh
\end{aligned}
\qquad
\begin{aligned}
& U(\varepsilon_B \cdot FUh) \cdot \eta_{UA} \\
= \; & \{ U \text{ a functor} \} \\
& U\varepsilon_B \cdot UFUh \cdot \eta_{UA} \\
= \; & \{ \eta \text{ natural} \} \\
& U\varepsilon_B \cdot \eta_{UB} \cdot Uh \\
= \; & \{ U\varepsilon \cdot \eta U = Id_U \} \\
& id_{UB} \cdot Uh \\
= \; & \{ id \} \\
& Uh .
\end{aligned}
$$

Finally we must show that $\varepsilon F \cdot F\eta = Id$ i.e. that for each $X \in \mathcal{X}$, the two arrows $\varepsilon_{FX} \cdot F\eta_X$ and id_{FX} from FX to FX are equal. In fact, they are both equal to $(\eta_X)\#$ because



$U(id_{FX}) \cdot \eta_X = id_{UFX} \cdot \eta_X = \eta_X$ and

$$
\begin{aligned}
& U(\varepsilon_{FX} \cdot F\eta_X) \cdot \eta_X \\
= \; & \{ U \text{ a functor} \} \\
& U\varepsilon_{FX} \cdot UF\eta_X \cdot \eta_X \\
= \; & \{ \eta \circ \eta \} \\
& U\varepsilon_{FX} \cdot \eta_{UFX} \cdot \eta_X \\
= \; & \{ \text{def of } U\varepsilon \cdot \eta U \} \\
& (U\varepsilon \cdot \eta U)_{FX} \cdot \eta_X \\
= \; & \{ U\varepsilon \cdot \eta U = Id_U \} \\
& id_{UFX} \cdot \eta_X \\
= \; & \{ id \} \\
& \eta_X .
\end{aligned}
$$

The adjunction we have constructed is unique:

* F's action on arrows is determined by the requirement that η be natural.

* ε is determined by the requirement that $U\varepsilon \cdot \eta U = Id_U$

□

This is the situation as we now see it: adjunctions

$$\langle F, U, \eta, \varepsilon \rangle : \mathcal{X} \longrightarrow \mathcal{A}$$

are in one-to-one correspondence with systems of universal arrows

$$\langle F_0 X, \eta_X \rangle$$

from $X \in \mathcal{X}$ to U, and such systems are unique up to isomorphism. We deduce that adjunctions are determined up to isomorphism by the functor U.

To put it precisely, if $\langle F, U, \eta, \varepsilon \rangle$ and $\langle F', U, \eta', \varepsilon' \rangle$ are two adjunctions sharing U, then there is a natural transformation $\theta : F \xrightarrow{\cdot} F'$ such that

$$\eta' = U\theta \cdot \eta$$
$$\varepsilon' \cdot \theta U = \varepsilon$$

and all the components θ_X are isomorphisms $FX \longrightarrow FX'$. In the notation of the proof above, $\theta_X = (\eta'_X)\#$.

Exercises III - again, these are premature.

(1) Closure operations as monads

A partially ordered set $\langle X, \leq \rangle$ may be turned into a category by saying that there is one arrow $f : x \to y$ for $x, y \in X$ if $x \leq y$, and none otherwise.

A closure operation on X is a function $x \longmapsto \overline{x}$ which is monotonic and satisfies $x \leq \overline{x}$ and $\overline{\overline{x}} \leq \overline{x}$. Show how a closure operation may be regarded as (the object part of the functor of) a monad.

(2) Non-deterministic "functions" as relations

One way of modelling non-deterministic programs is as binary relations between inputs and outputs. Another way is as functions from inputs to sets of outputs. Show that the definitions of composition in these two cases - as composition of binary

relations and as Kleisli composition - are equivalent.

(3) In Wadler's model of exceptions, a program returns the list $[a_1, a_2, \ldots]$.
Is it necessarily true that the same program in Spivey's model returns
Just a_1
?

(4) Functions with side-effects can be modelled as functions in
$X \longrightarrow (\text{state} \longrightarrow (Y \times \text{state}))$.
Composition is defined by

$$(g \odot f)\ x\ s = g\ y\ s' \quad \underline{\text{where}}\ (y, s') = f\ x\ s$$

This can be seen as the Kleisli composition for a certain monad.
Another definition is

$$(g \odot f)\ x\ s = g\ y\ s \quad \underline{\text{where}}\ (y, s) = f\ x\ s.$$

This is less easy to implement on a

conventional machine, but is it the Kleisli composition for any monad?

(5) Comparison Theorem for the Kleisli category

Let $\langle F, U, \eta, \varepsilon \rangle : \mathcal{X} \longrightarrow \mathcal{A}$ be an adjunction. Form the monad $\langle T, \eta, \mu \rangle$ and the Kleisli category

$$\langle F_T, U_T, \eta, \varepsilon_T \rangle : \mathcal{X} \longrightarrow \mathcal{X}_T .$$

Show how to define a functor

$$L : \mathcal{X}_T \longrightarrow \mathcal{A}$$

such that $F = LF_T$ and $U_T = UL$ and $\varepsilon L = L\varepsilon_T$.

6.4/ Examples of universals

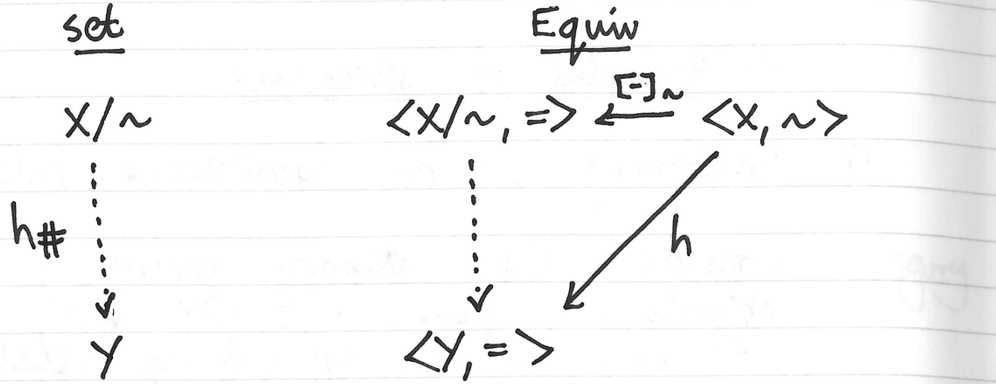
① Factoring out an equivalence relation.

Consider the category Equiv, in which objects are pairs $\langle X, \sim \rangle$ with X a set and $\sim$ an equivalence relation on X. The arrows are functions $h : X \to Y$ with $x \sim x' \Rightarrow hx \approx hx'$.

There's an obvious functor $G : \underline{\text{Set}} \to \text{Equiv}$ that maps X to $\langle X, =_X \rangle$ and $f : X \to Y$ to itself: $x = x' \Rightarrow fx = fx'$.

If $\langle X, \sim \rangle \in \text{Equiv}$, we can form the set $X/\sim$ of equivalence classes under $\sim$. There's a function $[-]_\sim : X \to X/\sim$ defined by $[x]_\sim = \{ y \mid x \sim y \}$. Actually, this function is an arrow $\langle X, \sim \rangle \longrightarrow \langle X/\sim, = \rangle$ because $x \sim y \Rightarrow [x]_\sim = [y]_\sim$.

In fact, $[-]_\sim$ is a universal arrow from $\langle X, \sim \rangle$ to G. The universality means that any arrow $h : \langle X, \sim \rangle \to \langle Y, = \rangle$ factors uniquely through $[-]_\sim$



There is a unique function $h\#: X/\sim \to Y$ such that $h = Gh\# \cdot [-]_\sim$

This justifies the usual practice of defining a function

$$f: X/\sim \to Y$$

by $f[x] = hx$, provided that $x \sim y \Rightarrow hx = hy$.

The adjunction at work here is

$$\langle G, F, \eta, \varepsilon \rangle : \underline{\text{Equiv}} \longrightarrow \underline{\text{Set}}$$

where $F: \underline{\text{Equiv}} \to \underline{\text{Set}}$ is defined by

$$F\langle X, \sim\rangle = X/\sim$$

and if $h: \langle X, \sim\rangle \to \langle Y, \approx\rangle$ then $Fh: X/\sim \to Y/\approx$ is defined by

$$(Fh)[x]_\sim = [hx]_\approx .$$

The unit is the function

$$\eta_{\langle X,\sim\rangle} : \langle X, \sim\rangle \longrightarrow \langle X/\sim, =\rangle$$

defined by $\eta_{\langle X,\sim\rangle}\, x = [x]_\sim$

The counit is "practically the identity": it is the function

$$\varepsilon_X : X/= \longrightarrow X$$

defined by $\varepsilon_X [x]_= = x$.

We say F is a "right adjoint" of G. G also has a left adjoint: it is the more prosaic functor

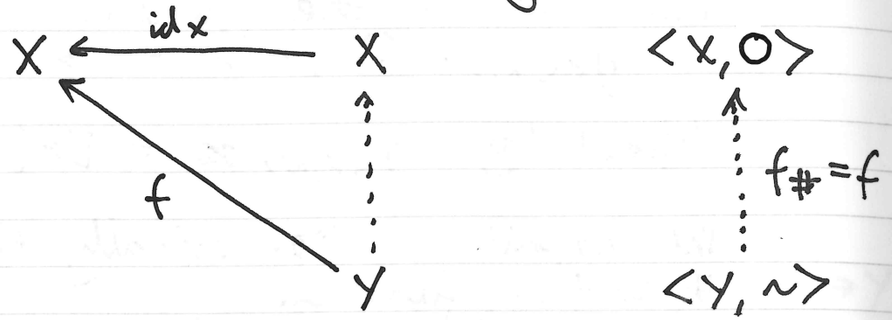
$$U: \underline{\text{Equiv}} \longrightarrow \underline{\text{Set}}$$

defined by

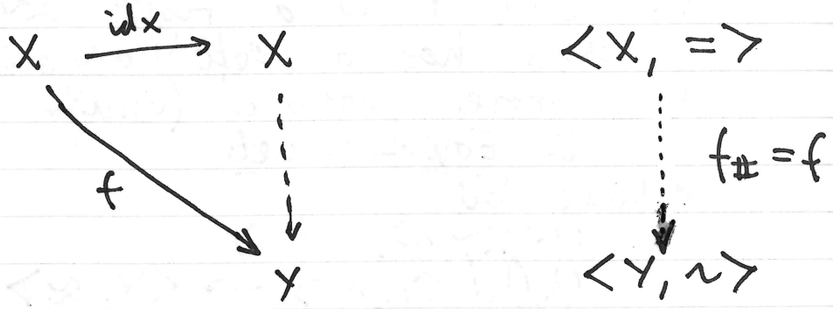
$$U\langle X, \sim\rangle = X$$
$$U(h: \langle X,\sim\rangle \to \langle Y, \approx\rangle) = h: X \longrightarrow Y.$$

Surprisingly, U also has a left adjoint, namely the functor $K: \underline{\text{Set}} \to \underline{\text{Equiv}}$ defined by $KX = \langle X, O\rangle$ where O is the universal relation with $x\, O\, y$ for every x and y. If $f: X \to Y$ then $Kf = f: \langle X, O\rangle \to \langle Y, O\rangle$ If $x\, O\, x'$ then certainly $fx\, O\, fx'$.

This last adjunction gives a universal arrow from U to any set X:

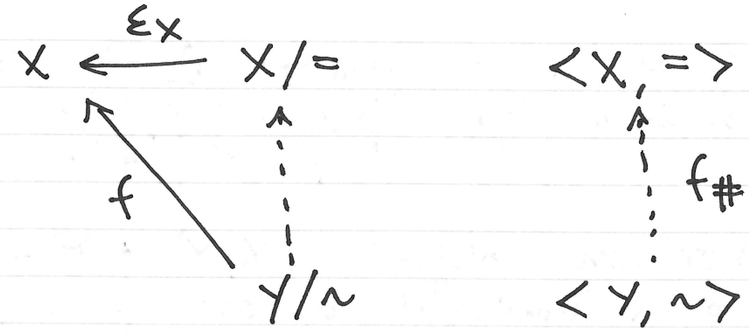


This captures the idea that O is the coarsest equivalence relation on X, just as the universal arrow



captures the idea that $=_X$ is the finest.

Finally, the following picture shows that ε_X is a universal arrow from F to X.



In this picture, f# is defined by

$$f^{\#}\, y = f([y]_\sim)$$

It is an arrow in Equiv because

$$y \sim y' \Rightarrow [y]_\sim = [y']_\sim \Rightarrow f^{\#}([y]_\sim) = f^{\#}([y']_\sim).$$

② Products and sums

If $\mathcal{C}$ is a category, we can form the category $\mathcal{C}^2$, in which the objects are pairs of objects of $\mathcal{C}$, and the arrows

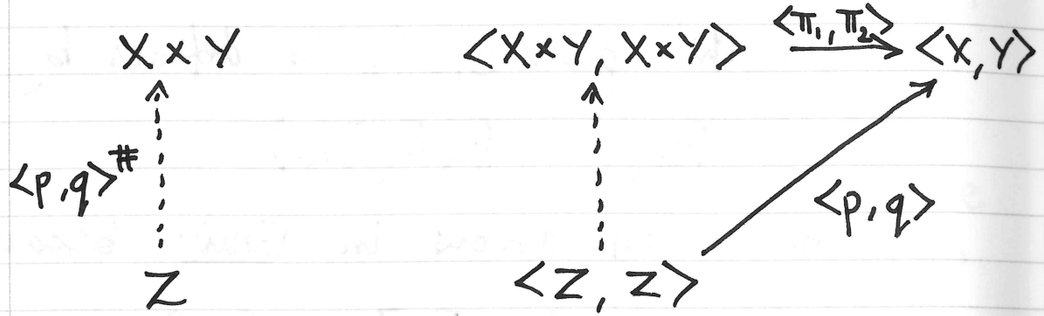
$$h : \langle X_1, X_2 \rangle \longrightarrow \langle Y_1, Y_2 \rangle$$

are pairs of arrows $h = \langle f, g \rangle$ where $f : X_1 \to Y_1$, $g : X_2 \to Y_2$ in $\mathcal{C}$.

Now suppose $\mathcal{C}$ has finite products.

There is a functor $\Delta : \mathcal{C} \to \mathcal{C} \times \mathcal{C}$ defined by
$\Delta X = \langle X, X \rangle$
$\Delta f = \langle f, f \rangle$

For any object $\langle X, Y \rangle \in \mathcal{C} \times \mathcal{C}$ there is a universal arrow from Δ to $\langle X, Y \rangle$



For any pair of functions $p : Z \to X$ and $q : Z \to Y$, there is a unique function $(p \| q) : Z \to X \times Y$ such that

$$\pi_1 \cdot (p \| q) = p$$

$$\pi_2 \cdot (p \| q) = q$$

i.e. $(p \| q)\, z = (p\, z,\ q\, z)$

If $\mathcal{C}$ has finite products, this works for all X, Y, and there is an adjunction $\langle \Delta, - \times -, \delta, \varepsilon \rangle : \mathcal{C} \to \mathcal{C} \times \mathcal{C}$

with $\delta_X : X \to X \times X$ defined by $\delta_X(x) = (x, x)$
and $\varepsilon_{\langle X,Y \rangle} : \langle X \times Y, X \times Y \rangle \to \langle X, Y \rangle$
defined by $\varepsilon_{\langle X,Y \rangle} = \langle \pi_1, \pi_2 \rangle$.

For sums, we get a universal arrow from $\langle X, Y \rangle$ to Δ and an adjunction
$\langle - \uplus -, \Delta, \nabla, \xi \rangle : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$
where
$\nabla_{\langle X,Y \rangle} = \langle \rho_1, \rho_2 \rangle : \langle X, Y \rangle \to \langle X \uplus Y, X \uplus Y \rangle$

$\xi_X : X \uplus X \to X$.

③ Exceptions

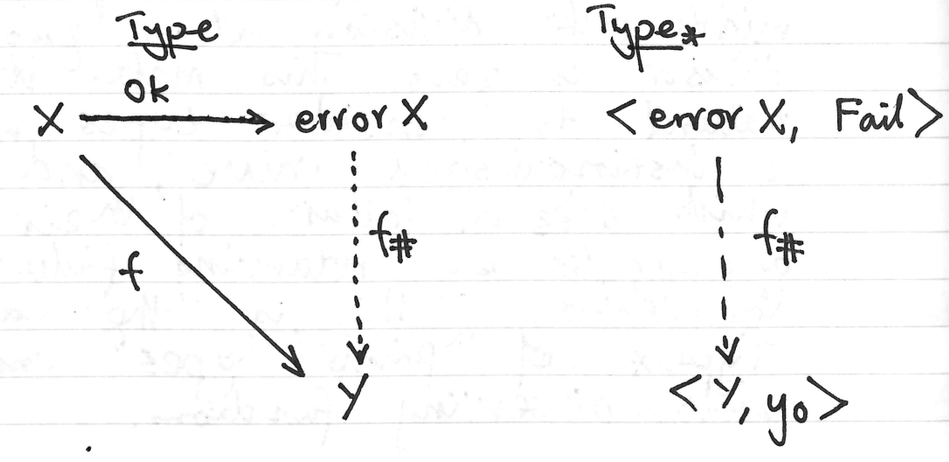
One way of dealing with computations that may fail is to have them return an error value: for example, we might make division return zero if the divisor is zero. This makes it natural to consider types with a distinguished value and functions which pass on failure of their arguments by returning failure themselves. This is the category $\mathbf{Type}_*$ of "pointed types" and point-preserving functions.

An object in Type$_*$ is a pair $\langle X, x_0 \rangle$, where X is a type and $x_0 : X$. An arrow $f : \langle X, x_0 \rangle \longrightarrow \langle Y, y_0 \rangle$ of Type$_*$ is a function $f : X \longrightarrow Y$ s.t. $f\, x_0 = y_0$.

The most conservative way of having a function return an error value is to make the error value quite separate from the normal values:

$$\text{error} \quad ::= \quad \text{Ok}\ \alpha \mid \text{Fail}$$

This type gives us a universal arrow from any type X to the forgetful functor U: Type$_*$ $\longrightarrow$ Type



Any function $f : X \longrightarrow Y$ can be written as $f_\# \cdot \text{Ok}$, where $f_\# : \langle \text{error } X, \text{Fail} \rangle \longrightarrow \langle Y, y_0 \rangle$ is defined by

$$\begin{aligned} f_\#\ (\text{Ok}\ x) &= f\, x \\ f_\#\ \text{Error} &= y_0 \end{aligned}$$

The adjunction at work here has as its unit the polymorphic constructor $\text{Ok}_X : X \longrightarrow \text{error } X$ and as its counit the "exception handler" $?\, x_0 : \langle \text{error} X, \text{Fail} \rangle \longrightarrow \langle X, x_0 \rangle$ defined by

$$\begin{aligned} (\text{Ok}\ x)\ ?\ x_0 &= x \\ \text{Fail}\ ?\ x_0 &= x_0. \end{aligned}$$

The functor F adjoint to U acts on an arrow $f : X \longrightarrow Y$ to give an arrow $f\bullet : \text{error } X \longrightarrow \text{error } Y$ defined by

$$\begin{aligned} f\bullet(\text{Ok}\ x) &= \text{Ok}\,(f\ x) \\ f\bullet\ \text{Fail} &= \text{Fail} \end{aligned}$$

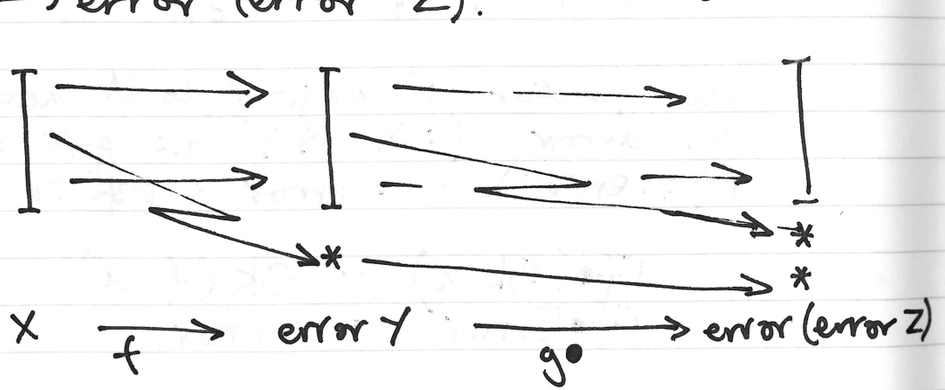
This function applies f to successful inputs, but passes on the failure of inputs that have already failed.

These functions provide some of the elements needed to simulate exceptions in a functional language that does not have them built in. Another element is composition of partial functions. If $f : X \longrightarrow \text{error } Y$ and $g : Y \longrightarrow \text{error } Z$, how can we "compose" them to get a function $g \odot f : X \longrightarrow \text{error } Z$?

The result type of f - error Y - does not match the argument type of g, but it does match with the function

$g\bullet : \text{error } Y \longrightarrow \text{error}(\text{error } Z)$

So we can form the function $(g\bullet) \cdot f$: $X \longrightarrow \text{error}(\text{error } Z)$.



The result type, error (error Z) contains two extra points: Fail, which corresponds to failure of f, and (Ok Fail), which corresponds to success of f followed by failure

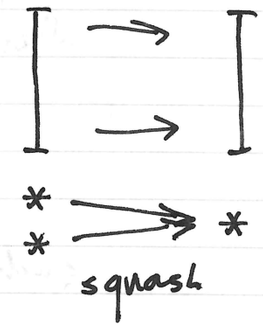
of g. But typically, we don't care about such a fine taxonomy of failure, so we should finish by merging these two error values into one — and incidentally mapping the success value (Ok (Ok x)) to a simple (Ok x). This is achieved by the function squash:

```
squash : error (error α) → error α
squash (Ok (Ok x)) = Ok x
squash (Ok x)      = Fail
squash Fail        = Fail
```

In fact, squash = (? Fail).

We define

$g \odot f = \text{squash} \,.\, (g\bullet) \,.\, f$.



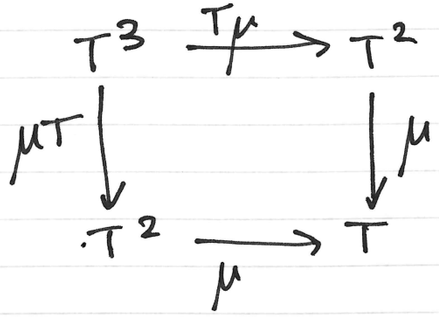
This is an example of Kleisli composition, a categorical construction we shall see more of later.

7/ Monads

We have been using adjunctions between Type and another category as a way of "specifying" data types. In this set-up all that really exists as a program is contained in the category Type. Monads provide a way of presenting the structure from an adjunction entirely within one category.

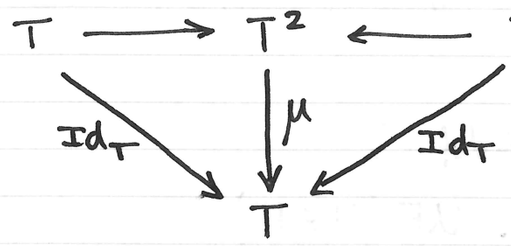
Definition Let $\mathcal{X}$ be a category. A **monad** on $\mathcal{X}$ is a triple $\langle T, \eta, \mu \rangle$ where T is a functor from $\mathcal{X}$ to $\mathcal{X}$
η is a natural transformation $Id_{\mathcal{X}} \xrightarrow{\bullet} T$
μ is a natural transformation $T^2 \xrightarrow{\bullet} T$
such that

$$\mu \cdot T\mu = \mu \cdot \mu T$$



"associativity"

$$\mu \cdot T\eta = Id_T = \mu \cdot \eta T$$



"identity"

Example $\langle$ list, $[-]$, $+\!\!+/\rangle$ is a monad on Type:

$$list : Type \longrightarrow Type \text{ is a functor}$$

$$[-] : Id_{Type} \xrightarrow{\bullet} list \text{ is a n.t.}$$

$$+\!\!+/ : list^2 \xrightarrow{\bullet} list \text{ is a n.t.}$$

and $+\!\!+/ \cdot (+\!\!+/)* = +\!\!+/ \cdot +\!\!+/$

$+\!\!+/ \cdot [-] = id_{list\,\alpha} = +\!\!+/ \cdot [-]*$

Proposition If $\langle F, U, \eta, \varepsilon \rangle : \mathcal{X} \longrightarrow \mathcal{A}$ then there is a monad $\langle T, \eta, \mu \rangle$ on $\mathcal{X}$ given by $T = UF$
$\mu = U\varepsilon F$.

Proof Plainly $T: \mathcal{X} \to \mathcal{X}$, $\eta: T = UF \dot{\to} Id_{\mathcal{X}}$ and $\varepsilon: Id_{\mathcal{U}} \dot{\to} FU$ $\therefore$ $\mu = U\varepsilon F: T^2 = U(FU)F \dot{\to} T = U(Id_{\mathcal{U}})F$.

Now $\mu . T\mu$

$= \{ \text{def } \mu, \text{def } T \}$

$U\varepsilon F . UF(U\varepsilon F)$

$= \{ \text{factorize} \}$

$U(\varepsilon . FU\varepsilon)F$

$= \{ \varepsilon \circ \varepsilon \}$

$U(\varepsilon . \varepsilon FU)F$

$= \{ \text{multiply out} \}$

$U\varepsilon F . (U\varepsilon F)UF$

$= \{ \text{def } T, \text{def } \mu \}$

$\mu . \mu T$

Also

$\mu . \eta T$	$\mu . T\eta$
$= \{ \text{def } \mu, \text{def } T \}$	$= \{ \text{def } T, \text{def } \mu \}$
$U\varepsilon F . \eta UF$	$U\varepsilon F . \eta$
$= \{ \text{factorize} \}$	$= \{ \text{factorize} \}$
$(U\varepsilon . \eta U)F$	$U(\varepsilon F . F\eta)$

$= \{ \text{triangle law} \}$	$= \{ \text{triangle law} \}$
$Id_U F$	UId_F
$= \{ T = UF \}$	$= \{ T = UF \}$
Id_T	Id_T

□

In fact, our first example <list, [-], ++> comes from this proposition, as do the following:

<error, Ok, squash>

<btree, Leaf, bgraft>

<tree, Tip, graft>

Here is an example that does not obviously come from the same source. Let $\langle Z, \oplus, e \rangle$ be a monoid and define

$TX = X \times Z$

$T(f: X \to Y) = f \text{ cross } id_Z : X \times Z \to Y \times Z$

$\eta_X(x) = (x, e)$

$\mu_X((x, z_1), z_2) = (x, z_1 \oplus z_2)$

We find that $\langle T, \eta, \mu \rangle$ is a monad: in fact, the law $\mu . T\mu = \mu . \mu T$ comes

directly from the associative law for $\oplus$, and the laws $\mu.T\eta = Id_T = \mu.\eta T$ come from the fact that e is a left and right identity element for $\oplus$.

So the question arises, whether all monads can be obtained as the images of adjunctions by our proposition. The answer is yes, and there are two constructions that give this answer: the Kleisli category, and the category of Eilenberg-Moore algebras.

7.1 The Kleisli category

Definition Let $T = \langle T, \eta, \mu \rangle$ be a monad on a category $\mathcal{X}$. The Kleisli category $\mathcal{X}_T$ of T has as objects the objects of X, but as arrows $f: X \to Y$ in $\mathcal{X}_T$ the arrows $f: X \to TY$ in $\mathcal{X}$. The identity arrows are $\eta_X : X \to X$ in $\mathcal{X}_T$, and composition of $f: X \to Y$ and $g: Y \to Z$ in $\mathcal{X}_T$ gives the arrow $g \odot f : X \to Z$ defined by

$$g \odot f = \mu_Z \cdot Tg \cdot f$$

Proposition The above does define a category.

Proof For example, we can show that the composition operator $\odot$ is associative: if $f: X \to Y$, $g: Y \to Z$, $h: Z \to W$ in $\mathcal{X}_T$, then

$$\begin{aligned}
&(h \odot g) \odot f \\
&= \{\text{def } \odot\} \\
&\mu_W \cdot T(\mu_W \cdot Th \cdot g) \cdot f \\
&= \{T \text{ a functor}\} \\
&\mu_W \cdot T\mu_W \cdot T^2 h \cdot Tg \cdot f \\
&= \{\text{associativity of } \mu\} \\
&\mu_W \cdot \mu_{TW} \cdot T^2 h \cdot Tg \cdot f \\
&= \{\mu \text{ natural}\} \\
&\mu_W \cdot Th \cdot (\mu_Z \cdot Tg \cdot f) \\
&= \{\text{def } \odot\} \\
&h \odot (g \odot f)
\end{aligned}$$

The identity arrows η_X are units for composition:

$$
\begin{array}{ll}
f \odot \eta_X & \eta_Y \odot f \\
= \{ \text{def } \odot \} & = \{ \text{def } \odot \} \\
\mu_Y \cdot Tf \cdot \eta_X & \mu_Y \cdot T\eta_Y \cdot f \\
= \{ \eta \text{ natural} \} & = \{ \text{identity law} \} \\
\mu_Y \cdot \eta_{TY} \cdot f & id_{TY} \cdot f \\
= \{ \text{identity law} \} & = \\
id_{TY} \cdot f & f . \\
= & \\
f & \square
\end{array}
$$

Example Take the monad we generated from a monoid $\langle Z, \oplus, e \rangle$. Arrows in the Kleisli category are functions $f : X \longrightarrow Y \times Z$, and the composition in the Kleisli category is defined by

$$
\begin{aligned}
(g \odot f)\, x &= (z, a \oplus b) \\
&\text{where } (y, a) = f\, x \\
&\qquad (z, b) = g\, y
\end{aligned}
$$

Imagine that Z is the type of character strings with $\oplus$ being string concatenation

and e the empty string. Then the type of $f : X \longrightarrow Y \times$ string is right for a function that, as well as producing a result in Y, prints an output string too. The Kleisli composition puts together the strings output by f and by g by concatenating them.

Proposition The Kleisli category $\mathcal{X}_T$ of a monad $\langle T, \eta, \mu \rangle$ on $\mathcal{X}$ is related to $\mathcal{X}$ by an adjunction

$$\langle F_T, U_T, \eta_T, \mu_T \rangle : \mathcal{X} \longrightarrow \mathcal{X}_T$$

where $F_T X = X \in \mathcal{X}_T$

$F_T (f : X \to Y) = \eta_Y \cdot f : FX \to FY.$

$U_T X = TX$

$U_T (h : X \to Y) = \mu_Y \cdot Th : U_T X \to U_T Y$

$\eta_T = \eta$

$(\varepsilon_T)_X = id_{TX} : TX \longrightarrow X$ in $\mathcal{X}_T$.

Furthermore, the monad induced on $\mathcal{X}$ by this adjunction is T.

Proof We prove the two triangle laws:

$$
\begin{aligned}
&(\varepsilon_T F_T \odot F_T \eta_T)X \\
&= \{\text{def}\} \\
&\quad id_{TX} \odot (\eta_{TX} \cdot \eta_X) \\
&= \{\text{def } \odot\} \\
&\quad \mu_X \cdot T\, id_{TX} \cdot \eta_{TX} \cdot \eta_X \\
&= \{\text{id}\} \\
&\quad \mu_X \cdot \eta_{TX} \cdot \eta_X \\
&= \{\text{identity law}\} \\
&\quad \eta_X
\end{aligned}
$$

η_X ← which is the identity $X \to X$ in $\mathcal{K}_T$

$$
\begin{aligned}
&(U_T \varepsilon_T \cdot \eta_T U_T)X \\
&= \{\text{def}\} \\
&\quad \mu_X \cdot T(id_{TX}) \cdot \eta_{TX} \\
&= \\
&\quad \mu_X \cdot \eta_{TX} \\
&= \\
&\quad \cdot\, id_{TX}.
\end{aligned}
$$

For the monad induced on $\mathcal{K}$, we calculate

$$U_T F_T X = U_T X = TX$$
$$U_T F_T f = \mu_Y \cdot T(\eta_Y \cdot f) = Tf$$
$$(U_T \varepsilon_T F_T)X = \mu_X \cdot T(id_{TX}) = \mu_X$$

so it is the same as T. □

Although the Kleisli construction typically reconstructs from a monad an adjunction different from the one we originally used to induce it, this new adjunction is often meaningful and useful.

For example, the arrows in the Kleisli category for the monad <tree, Tip, graft> are functions $f: X \to \text{tree } Y$. We can think of them as substitutions of expressions in the variables Y for the variables X. Kleisli composition is composition of substitutions, and the functor U_T: Type$^{\text{tree}} \to$ Type shows how substitution act on expressions: $U_T f$: tree $X \to$ tree Y. The fact that U_T is a functor tells us that composing substitutions composes their actions on expressions.

7.2 Eilenberg-Moore algebras

Definition Let $T = \langle T, \eta, \mu \rangle$ be a monad on $\mathcal{X}$. The category $\mathcal{X}^T$ of Eilenberg-Moore algebras has as objects the algebras $\langle X, f \rangle$ where $X \in \mathcal{X}$ and $f : TX \longrightarrow X$ is such that

$$f \cdot \eta_X = id_X$$

$$\begin{array}{ccc} X & \xrightarrow{\eta_X} & TX \\ & \searrow^{id_X} & \downarrow f \\ & & X \end{array} \qquad \begin{array}{ccc} T^2X & \xrightarrow{Tf} & TX \\ \downarrow \mu_X & & \downarrow f \\ TX & \xrightarrow[f]{} & X \end{array}$$

and $f \cdot Tf = f \cdot \mu_X$.

The arrows $h : \langle X, f \rangle \longrightarrow \langle Y, g \rangle$ are functions $h : X \longrightarrow Y$ such that

$$h \cdot f = g \cdot Th$$

$$\begin{array}{ccc} TX & \xrightarrow{Th} & TY \\ \downarrow f & & \downarrow \\ X & \xrightarrow{h} & Y \end{array}$$

Example Take the monad $\langle list, [-], +\!\!+/ \rangle$. If $\langle X, \oplus, e \rangle$ is a monoid, then one example of an Eilenberg-Moore algebra is $\langle X, \oplus/ \rangle$. The two requirements are

$$\oplus/\,[x] = x$$
$$\oplus/ \cdot (\oplus/)* = \oplus/ \cdot +\!\!+/$$

which we recognize. In fact, this example, as we shall see, exhausts the range of algebras.

Proposition The category $\mathcal{X}^T$ of T-algebras of a monad $\langle T, \eta, \mu \rangle$ on $\mathcal{X}$ is related to $\mathcal{X}$ by an adjunction

$$\langle F^T, U^T, \eta^T, \varepsilon^T \rangle : \mathcal{X} \longrightarrow \mathcal{X}^T$$

which induces on $\mathcal{X}$ the monad T. This adjunction is given by

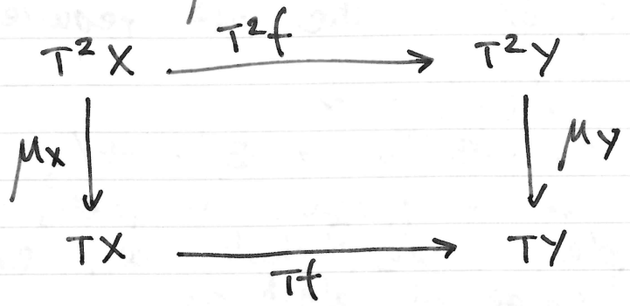
$$F^T X = \langle TX, \mu_X \rangle$$
$$F^T(f : X \to Y) = Tf : \langle TX, \mu_X \rangle \to \langle TY, \mu_Y \rangle$$

$$U^T \langle X, f \rangle = X$$
$$U^T(h : \langle X, f \rangle \to \langle Y, g \rangle) = h : X \to Y$$

$$\eta^T = \eta$$

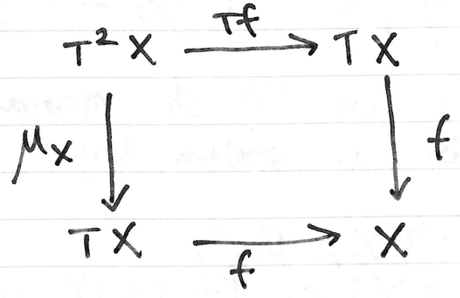
$$\varepsilon^T_{\langle X, f \rangle} = f : \langle TX, \mu_X \rangle \longrightarrow \langle X, f \rangle.$$

Proof (A) If $f: X \to Y$ then
$Tf: \langle TX, \mu_X \rangle \to \langle TY, \mu_Y \rangle$ in $\mathcal{X}^T$
because μ is natural:



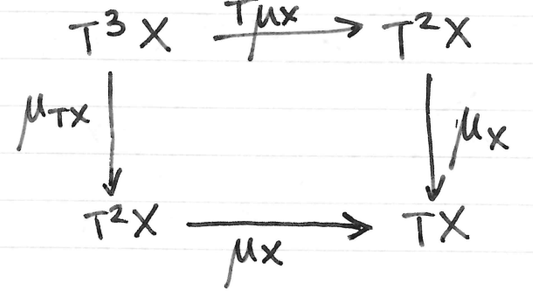
So F^T is a functor $\mathcal{X} \to \mathcal{X}^T$.
U^T is clearly a functor, and $\eta^T = \eta$
is clearly natural $Id_{\mathcal{X}} \to T = U^T F^T$.

If $\langle X, f \rangle \in \mathcal{X}^T$ then $f: \langle TX, \mu_X \rangle \to \langle X, f \rangle$

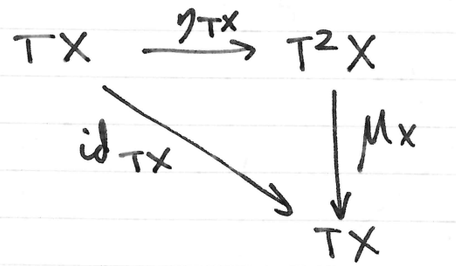


by a defining condition for $\langle X, f \rangle$ to be a T-algebra.

(A) $\langle TX, \mu_X \rangle$ is a T-algebra because

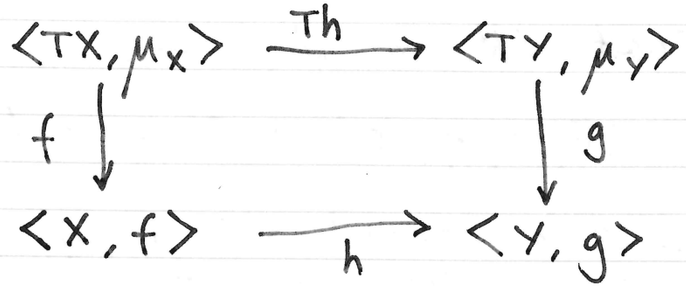


(T is a monad)



(another part of T's being a monad)

ε is natural



(h is a homomorphism)

The triangle laws: $\varepsilon F^T \cdot F^T \eta^T = Id_{F^T}$

$$\begin{array}{ccc} \langle TX, \mu_X \rangle & \xrightarrow{T\eta_X} & \langle T^2X, \mu_{TX} \rangle \\ & \searrow^{id_{TX}} & \downarrow \mu_X \\ & & \langle TX, \mu_X \rangle \end{array}$$

$U^T \varepsilon^T \cdot \eta^T U^T = Id_{U^T}$

$$\begin{array}{ccc} X & \xrightarrow{\eta_X} & TX \\ & \searrow^{id_X} & \downarrow f \\ & & X \end{array}$$

(another part of f's being an algebra)

7.3 Comparison with algebras

Start with an adjunction

$$\langle F, U, \eta, \varepsilon \rangle : \mathcal{H} \longrightarrow \mathcal{A}$$

Make a monad $\langle T = UF, \eta, \mu = U\varepsilon F \rangle$ on $\mathcal{H}$. Take the category of T-algebras, related by another adjunction to $\mathcal{H}$:

$$\langle F^T, U^T, \eta^T, \varepsilon^T \rangle : \mathcal{H} \longrightarrow \mathcal{H}^T$$

How is this adjunction related to

the one we started with?